תכנון אלגוריתמים	18.10.2020

הרצאה 1: פתרון בעיות ע"י רדוקציות

Textbook: Cormen, Leiserson, Rivest, Stein. Introduction to Algorithms.

מרצה: אורי שטמר

הקדמה

שמות המרצים: עמוס ביימל, סבסטיאן בן דניאל, עדן כלמטץ', יעל שטיין, אורי שטמר, מירב זהבי

שמות המתרגלים: תום הס, שחף פינדר, גיא סער, בנימין ברנד, מני סדיגורסקי, עידן אטיאס, אור נחמני, יהונתן גבאי, אריאל אלפרין, טל אוחיון, מוראל סיאני, יעל אזולאי, עירית שלי, שלומי ויצמן, יהושע גרוגין

תגבורים: עוזי פרידמן

אתר הקורס: www.cs.bgu.ac.il/~algo211

בעיות/שאלות מנהלתיות: מערכת פניות סטודנטים דרך אתר הקורס

ספרים:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms.
2. J. Kleinberg and E. Tardos, Algorithm Design.

מטרת הקורס: פיתוח חשיבה אלגוריתמית
- להגדיר בצורה מדוייקת מה רוצים לפתור
- למצוא אלגוריתם נכון ויעיל שפותר את הבעיה
- הוכחת נכונות של האלגוריתם

איך נפתח חשיבה אלגוריתמית? נראה מגוון של בעיות ונתכנן אלגוריתמים יעילים שפותרים אותם. נלמד כמה טכניקות שונות לפיתוח אלגוריתמים.

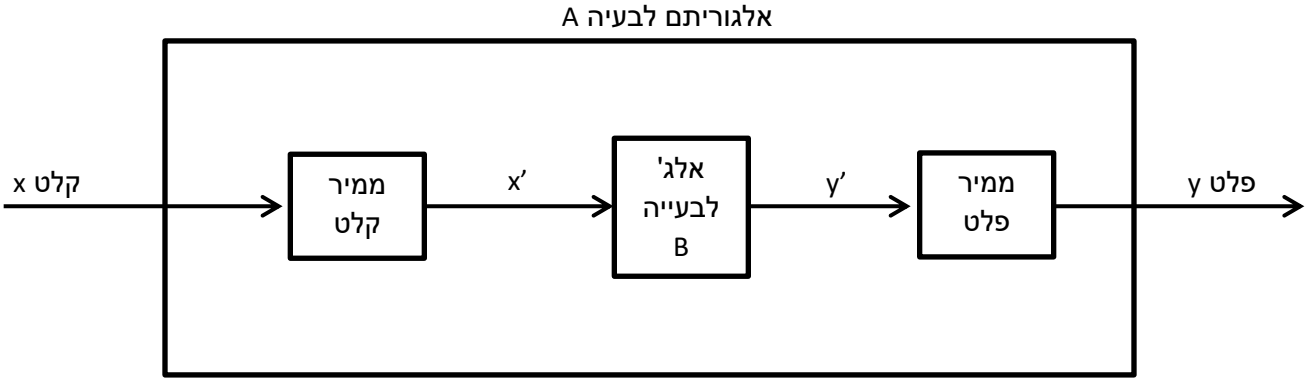
דוגמאות לטכניקות שנלמד:
- פתרון בעיות ע"י רדוקציות: פתרון בעיה חדשה ע"י אלגוריתם לבעיה ישנה.
- אלגוריתמים חמדנים: אלו אלגוריתמים שבכל שלב מבצעים פעולה שנראית הכי טובה באותו רגע
- אלגוריתמים דינאמיים: שיטה לקחת אלגוריתם רקורסיבי לא יעיל ולהפוך אותו לאלגוריתם יעיל

דגשים בקורס:
- בהינתן בעיה, נתחיל מלשאול את עצמנו איך אפשר לפתור אותה בצורה נאיבית (לא בהכרח ע"י אלגוריתם יעיל). המטרה בכך היא לקבל אינטואיציה על הבעיה.
- חשיבה מלמעלה למטה (Top Down)
  - נתחיל מתאור ברמה גבוהה/בצורה מופשטת של האלגוריתם (למשל, לא נפרט בהתחלה באיזה מבנה נתונים ספציפי נשתמש)
  - הוכחת נכונות של האלגוריתם ברמה הגבוהה
  - לאחר מכן נראה איך לממש את האלגוריתם בצורה יעילה וננתח סיבוכיות זמן ריצה
- הוכחת נכונות בגישת Top Down
  - נתחיל בכתיבת משפט שמבטיח את נכונות האלגוריתם
  - נכתוב טענות עזר שישמשו להוכחת המשפט

- נוכיח את המשפט בעזרת טענות העזר
- נוכיח את טענות העזר

רדוקציות – פתרון בעיה חדשה בעזרת אלג' ידוע לבעיה ישנה

אנחנו רוצים לפתור בעיה A ויש לנו פתרון לבעיה B. בבעיה A יש לנו קלט x ואנחנו צריכים להחזיר פלט y. ניקח את הקלט x וע"י אלגוריתם ראשון נמיר אותו ל- x' על מנת שיהיה קלט לאלג' הפותר את בעיה B. את הפלט של האלג' הזה, y', נמיר ל- y.



- רדוקציה מורכבת משני אלגוריתמים:
  - ממיר קלט
  - ממיר פלט

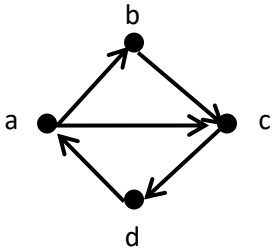
- אלגוריתם שמבוסס על רדוקציה:
  - מריץ ממיר קלט
  - מריץ את האלגוריתם לפתרון B
  - מריץ את ממיר הפלט

צעד אחורה: הגדרת בעייה

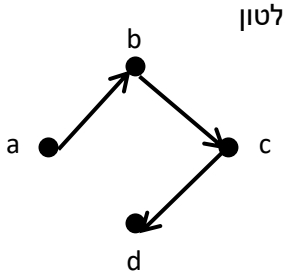
מופע: הגדרת קלט חוקי לבעיה (לדוגמה: גרף מכוון)

פתרון: הגדרה מתי פלט מסויים הוא חוקי לבעיה (לדוגמה: מסלול פשוט העובר דרך כל צומת בגרף בדיוק פעם אחת. מסלול כזה נקרא מסלול המילטון). לא בהכרח קיים פתרון חוקי.

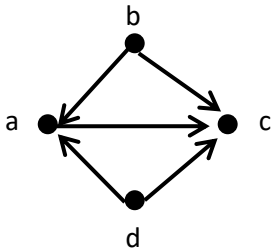
דוגמה: עבור הגרף



המסלול הבא מהווה מסלול המילטון



דוגמה נוספת: עבור הגרף הבא לא קיים פתרון חוקי

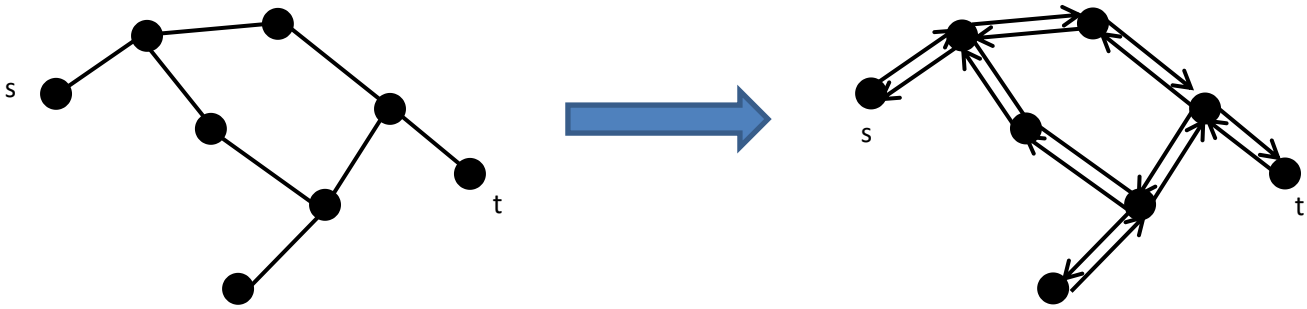


אלגוריתם לבעיה: אוסף הוראות, צעדים, שפותרים את הבעיה או אומרים שאין פתרון. כלומר, <u>לכל קלט</u> מחזירים פתרון חוקי אם קיים ואחרת אומרים "אין פתרון".

<u>דוגמה פשוטה לרדוקציה:</u>

נתון אלג' יעיל שמוצא מסלול קצר ביותר בין זוג קוד' בגרף <u>מכוון</u>. איך נוכל להשתמש באלג' זה כדי למצוא מסלול קצר ביותר בין זוג קוד' בגרף <u>לא-מכוון</u>?

נחליף כל קשת בגרף הלא מכוון בשתי קשתות מכוונות לשני הכיוונים:



<u>ממיר הקלט:</u> לוקח גרף לא-מכוון $G=(V,E)$ וזוג קוד' $s,t \in V$. מחזיר גרף מכוון $G'=(V,E')$ וזוג קוד' $s,t \in V$, כאשר

$$E' = \{(u,v),(v,u) : (u,v) \in E\}$$

<u>ממיר הפלט:</u> מחזיר את הפלט שהוחזר מהאלגוריתם (כלומר ממיר הפלט לא עושה כלום במקרה זה).

<u>האלגוריתם:</u>

(א) הרץ את ממיר הקלט, כלומר בהינתן G,s,t קבל G',s,t

(ב) מצא מסלול קצר ביותר מ- s ל- t ב- G'

(ג) החזר את המסלול הנ"ל, או החזר "לא קיים" אם לא קיים מסלול כנ"ל.

משפט: האלגוריתם מחזיר מסלול קצר ביותר מ- s ל- t ב- G (אם קיים).

לא נוכיח את המשפט (תראו דברים דומים בתרגול). אם היינו רוצים להוכיח עכשיו את המשפט, היינו יכולים להשתמש בטענת העזר הבאה (ואחכ להוכיח אותה):

טענת עזר: P הוא מסלול באורך ℓ מ- s ל- t ב- G אם ורק אם P הוא מסלול באורך ℓ מ- s ל- t ב- G'.

<u>ניתוח זמן ריצה:</u>
(א) ממיר הקלט: כל קשת משכפלים פעמיים $O(|V|+|E|)$
(ב) ניתן למצוא מסלול קצר ביותר ע"י BFS בזמן $O(|V|+|E|)$
(ג) ממיר הפלט: $O(1)$

סה"כ: $O(|V|+|E|)$

הגדרה פורמלית ל- "רדוקציה"

תהיינה A ו- B זוג בעיות נתונות. רדוקציה מבעיה A לבעיה B היא זוג פונקציות f, g כך ש:
- f היא פונקצית המרת קלט, המעבירה מופע של בעיה A למופע של בעיה B
- g היא פונקצית המרת הפלט, המעבירה קלט של בעיה A ופתרון של בעיה B לפתרון של בעיה A
- עבור מופע a לבעיה A, אם b הוא פתרון עבור המופע $f(a)$ תחת בעיה B אזי $g(x,b)$ הוא פתרון למופע a תחת בעיה A. (הגדרת נכונות)

כדי להוכיח את נכונות הרדוקציה, יש להוכיח שהאלגוריתם הבא פותר את בעיה A:
1. עבור מופע a לבעיה A, חשב $f(a)$
2. עבור המופע $f(a)$ לבעיה B, חשב פתרון b
3. חשב את $g(x,b)$ להיות הפתרון ל- a עבור בעיה A

שימו לב: כאשר מנתחים זמן ריצה של אלגוריתם מבוסס רדוקציה יש לשים לב לגודל הקלט שאנחנו מזינים לאלגוריתם שפותר את בעיה B (כלומר יש לפרט מהו גודלו של $f(a)$).

בעיית תתי המחרוזות

<u>קלט:</u> אוסף של n מחרוזות כל אחת באורך 3
דוגמה: GUR, URI, NGU, BEN, ION, ENG, RIO

<u>פלט חוקי:</u> מחרוזת באורך $n+2$ כך שהמחרוזות בקלט הן בדיוק כל תתי המחרוזות שלה באורך 3.
בדוגמה BENGURION

<u>הנחה:</u> כל מחרוזת בקלט מופיעה פעם אחת. כלומר, אף מחרוזת לא מופיעה פעמיים.

<u>דרך אחרת להסתכל על הבעיה:</u> מישהו החזיק מחרוזת באורך $n+2$ וגזר ממנה את כל תתי המחרוזות באורך 3 ונתן לנו את תתי המחרוזות בסדר כלשהו.

<u>פתרון פשוט ולא יעיל לבעיה:</u>
נבדוק את כל $n!$ האפשרויות לסדר את המחרוזות בקלט.

מתי סידור הוא אפשרי?

- אם שתי האותיות האחרונות במחרוזת במקום ה i הם שתי האותיות הראשונות במחרוזת במקום ה $i+1$

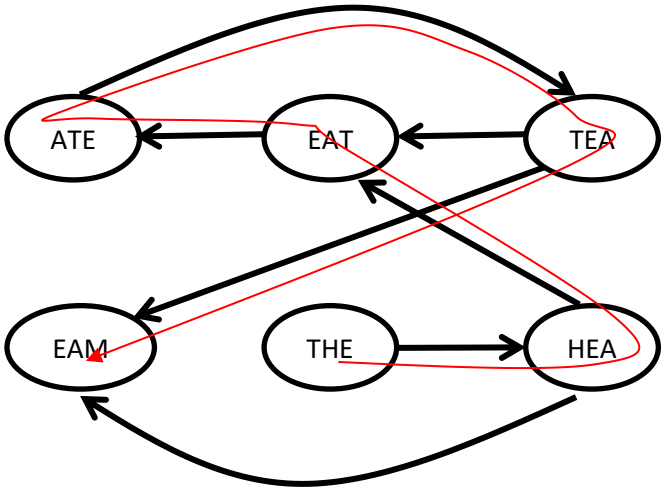
האלגורים הזה הוא מאוד לא יעיל: $n! = 2^{O(n \cdot \log n)}$

איך נתכנן אלגוריתם אחר?

<u>הגדרה:</u> מחרוזת w_j יכולה להופיע אחרי מחרוזת w_i אם שתי האותיות האחרונות של w_i הן שתי האותיות הראשונות של w_j.

נייצג את המחרוזות שיכולות להופיע אחת אחרי השניה בעזרת גרף.

<u>דוגמה:</u> *ATE, EAT, TEA, EAM, THE, HEA*



THEATEAM

בעצם אנחנו באותה בעיה מהדוגמה הראשונה של השיעור:
רוצים למצוא מסלול פשוט העובר דרך כל קוד' בגרף בדיוק פעם אחת.

<u>ממיר הקלט:</u> <u>קלט:</u> $w_1, w_2, \dots, w_n$

<u>פלט:</u> גרף מכוון $G = (V, E)$ כאשר $V = \{w_1, \dots, w_n\}$ וכאשר יש קשת מכוונת מ- w_i ל- w_j אם w_j יכולה להופיע אחרי w_i

<u>ממיר הפלט:</u> <u>קלט:</u> מסלול המילטון בגרף $w_{i_1}, w_{i_2}, \dots, w_{i_n}$.

נבנה מחרוזת באורך $n+2$ על ידי לקיחת w_{i_1} והוספת האות האחרונה ב- w_{i_2}, אח"כ האות האחרונה ב- w_{i_3}, וכן הלאה עד האות האחרונה ב- w_{i_n}.
(אם האלגוריתם החזיר שאין מסלול המילטון אז נחזיר שאין פתרון)

הראינו רדוקציה מבעיית תתי המחרוזות לבעיית מסלול המילטון.

האלגוריתם שנובע מהרדוקציה:

- הרץ ממיר קלט, כלומר בנה את הגרף G
- מצא מסלול המילטון ב- G
- הרץ ממיר פלט ומצא את המחרוזת. אם אין מסלול המילטון אז החזר "אין פתרון".

משפט: אם יש פתרון אז האלגוריתם יחזיר פתרון. אם אין פתרון אז האלגוריתם יחזיר שאין פתרון.

טענה 1: אם אין פתרון אז אין מסלול המילטון בגרף

טענה 2: אם יש פתרון לבעיה אז בגרף קיים מסלול המילטון. יתרה מכך, המחרוזת שממיר הפלט בונה ממסלול המילטון היא פתרון לבעיה.

נראה קודם כי 2 הטענות גוררות את המשפט:

* אם יש פתרון, אז עפ"י טענה 2 יש מסלול המילטון בגרף ולכן האלגוריתם שמוצא מסלול המילטון ימצא מסלול המילטון בגרף. עפ"י טענה 2, המחרוזת שנבנה ממנה היא פתרון.
* אם אין פתרון אז עפ"י טענה 1 אין מסלול המילטון. לכן האלגוריתם עבור בעיית מסלול המילטון יחזיר שאין מסלול ולכן האלגוריתם שבנינו יחזיר שאין פתרון.